BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian ini merujuk pada berbagai studi-studi terdahulu yang telah dilakukan dalam bidang analisa sentimen dan telah dicapai sebelumnya terkait dengan analisa sentimen yang akan dijadikan acuan penulis pada penelitian yang akan dilakukan:

Tabel 2. 1 Tabel Penelitian Terdahulu

Penulis	Judul	Jurnal	Metode	Hasil
Ramadhan, Z. F., & Mutiara, A. B. (2023)	Sentiment Analysis of Honkai: Star Rail Indonesian Language Reviews on Google Play Store Using Bidirectional Encoder Representations from Transformers Method	*International Journal of Engineering Science and Information Technology* 3(3):1-6 [22]	BERT	Akurasi model adalah 81%, dengan presisi yang bervariasi untuk kategori sentimen.
Setyani, S., Prasetiyowati, S. S., & Sibaroni, Y. (2023)	Multi Aspect Sentiment Analysis of Mutual Funds Investment App Bibit Using BERT Method	*International journal on information and communication technology* Vol. 9, No. 1, Jun2023. pp. 44-56 [23]	BERT dengan model pretrained IndoBERT	Akurasi 0,92, dan Skor F1 rata-rata: 0,73 di tiga aspek.
Maulidan, M. D., Sumarlinda, S., & Sopingi, S. (2024)	Development of Sentiment Analysis System of Simple Pol Application on Google Play Store Using Naive Bayes Classifier Method and BERT Prediction	*Journal of Dinda Data Science Information Technology and Data Analytics*, 4(2), 115–122 [25]	Pengklasifikasian Naïve Bayes dengan metode BERT untuk prediksi sentimen	Akurasi 88,7%, *precision* 88,5%, recall 100%.
Jojoa, M., Eftekhar, P., Nowrouzi-Kia (2022)	Natural language processing analysis applied to COVID-19 open-text opinions using a distilBERT model for sentiment categorization	*AI and Society* [26]	*DistilBERT* untuk teks opini	Akurasi: 0.823, Precision: 0.826, Recall: 0.793 and F1 Score: 0.803
Reza, R. F., Thoriq, M., & Millah, R. I. S. (2024)	Sentiment Analysis of Marketplace Review with IslamicPerspective using Fine-Tuning DistilBERT	*Khazanah Journal of Religion and Technology* [27]	DistilBERT berbahasa Indonesia	Akurasi: 0.8925, f1: 0.922303

Penulis	Judul	Jurnal	Metode	Hasil
Wu, Y., Jin, Z., Shi, C., Liang, P., & Zhan, T. (2024)	Research on the application of deep learning-based BERT model in sentiment analysis	*Applied and Computational Engineering, 67(1), 280–286* [28]	Model DistilBERT dengan parameter pra-terlatih	•Menunjukkan efektivitas Distilbert dalam tugas klasifikasi sentimen. •Menyoroti potensi transformatif model BERT dalam analisis sentimen.
Kofi Akpatsa, Samuel Lei, Hang Li, Xiaoyu Kofi Setornyo Obeng, Victor-Hillary Mensah Martey, (2022)	Online News Sentiment Classification Using DistilBERT	*Journal of Quantum Computing* 4(1), 1-11 [29]	DistilBERT	Model dapat berfungsi dengan baik dimana akurasi mencapai 94%
V. Pramanik and M. Maliha (2022)	Analyzing Sentiment Towards a Product using DistilBERT and LSTM	*International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), Greater Noida, India, 2022, pp.* 811-816 [20]	LSTM dan Distilbert untuk analisa sentimen	Distilbert mengungguli LSTM dalam metrik evaluasi
Nabila Rizky Amalia Putri, Trimono, Aviolla Terza Damaliana (2023)	Sentiment Analysis on Digital Korlantas POLRI Application Reviews Using the Distilbert Model	*Journal of Renewable Energy, Electrical, and Computer Engineering* [30]	DistilBERT	Akurasi 88% dengan ratio dataset 80:20
Willa Fatika Sari, Rida Rahim (2023)	Analisis Sentiment Review Pengguna BCA Mobile menggunakan teks mining	Repositori IMWI [31]	Naïve Bayes, 2 label positif dan negatif	Akurasi: 82%
A.Tikaningsih, P. lestari, A. Nurhopipah et al	Optuna Based Hyperparameter Tuning for Improving the Performance Prediction Mortality and Hospital Length	Telematika [32]	*Hyperparameter* dengan Optuna	Implementasi dari *hypertune* optimisasi menggunakan Optuna dapat meningkatkan

Penulis	Judul	Jurnal	Metode	Hasil
	of Stay for Stroke Patients			meningkatkan performa model.
P. Mahira, S. Taufik Edy, et al	Studi Empiris Model BERT dan DistilBERT Analisis Sentimen pada Pemilihan Presiden Indonesia	Indonesian Journal of Computer Science [33]	BERT dan DistilBERT	DistilBERT jauh lebih efisien dan cepat dibandingkan dengan BERT
Raymond Oetama (2021)	Sentiment Analysis about Indonesian Lawyers Club Television Program Using K-Nearest Neighbor, Naïve Bayes Classifier, and Decision Tree	*IJNMT (International Journal of New Media Technology)* [34]	K-NN, Naïve Bayes dan Decision tree	Akurasi di ketiga algoritma tidak lebih baik diangka 70% keatas
Dinar Ajeng Kristiyanti(2022)	Text Mining of PeduliLindungi Application Reviews on Google Play Store	*Faktor Exacta* [35]	Naïve Bayes	Akurasi 85%

Tabel 2.1 di atas menunjukkan berbagai penelitian sebelumnya yang berfokus mengenai analisa sentimen dengan beragam pendekatan metode dimulai dari penggunaan model KNN, Naïve Bayes, LSTM, Decision Tree, BERT dan DistilBERT. Pada penelitian ini dilakukan menggunakan metode yang telah digunakan sebelumnya, namun dengan objek dan studi kasus penelitian yang berbeda. Penelitian ini menggunakan objek data ulasan pengguna aplikasi BCA Mobile yang ada di platform Google Play Store yang bertujuan untuk mengklasifikasikan ulasan data berdasarkan sentimen. Penelitian yang sudah dilakukan [23], [25] memberikan hasil yang baik mengenai analisa sentimen menggunakan BERT dengan pretrained INDOBert namun pada kali ini peneliti akan menerapkan model lain BERT yakni model versi ringan yang dilakukan pada penelitian [26], [27], [28] dimana pada penelitian tersebut mengatakan bahwa DistilBERT juga mampu memberikan hasil yang serupa dengan BERT meskipun dengan jumlah layer yang lebih sedikit dibandingkan BERT dikarenakan proses distilasi serta ringan untuk digunakan [33].

Algoritma seperti KNN, Naïve Bayes, LSTM, dan Decision tidak digunakan dalam penelitian [31], [34], [35] karena secara keseluruhan *transformer model* seperti [30] unggul dalam *segi accuracy* terutama dalam klasifikasi sentimen positif

dan negatif serta dapat memahami polaritas teks lebih baik. IndoBERT ataupun IndoBERT-lite merupakan pretrained model yang memiliki keunggulan untuk Bahasa Indonesia, namun DistilBERT-*multilingual* memiliki keunggulan tambahan yakni kecepatan inferensi DistilBERT lebih tinggi 60% lebih ringan dibandingkan model BERT biasa yang nantinya penting untuk proses deployment pada web prototype Streamlit. Studi lain mengenai perbandingan fine-tune transformers *multilingual* menunjukkan bahwa DistilBERT-multilingual mampu menyamai atau bahkan melampaui IndoBERT-lite dalam tugas klasifikasi sentimen [36]. Pada akhirnya, fine-tuning DistilBERT-multilingual ini dipilih dan digunakan dalam penelitian ini karena IndoBERT secara umum hanya mensupport bahasa Indonesia saja sedangkan dalam penelitian ini nantinya akan menggunakan teks yang mendukung *multilingual*. Penelitian selanjutnya tentang pencarian parameter seperti yang telah dilakukan [32] akan digunakan karena berdasar pada penelitian tersebut Optuna dapat meningkatkan performa model dengan cara mencari parameter optimal dan terbaik secara otomatis.

Adapun kebaruan dari penelitian ini adalah terletak di pembuatan model analisa sentimen DistilBERT yang akan menggunakan model arsitektur *DistilBERT-multilingual- cased* yang mendukung lebih dari 1 bahasa yaitu Bahasa Indonesia dan Bahasa English, *fine tuning,* dan *hyperparameter tuning* menggunakan Optuna. Hasil *evaluasi model* dengan akurasi terbaik juga akan digunakan dalam klasifikasi sentimen dan diuji pada *website testing* singkat.

2.2 Tinjauan Teori

2.2.1 Ulasan

Ulasan, atau sering disebut juga dengan *review*, adalah teks yang berisi penjelasan, penafsiran, atau penilaian terhadap suatu karya seperti film, buku, musik, atau produk lainnya. Tujuan utama dari ulasan adalah untuk memberikan informasi yang komprehensif tentang suatu karya kepada pembaca, membantu mereka memahami isi dan kualitas karya tersebut, serta memberikan penilaian subjektif dari penulis ulasan [37]. Dalam konteks penelitian ini, ulasan merujuk pada penilaian dan komentar yang diberikan oleh pengguna terhadap suatu aplikasi yang tersedia pada

platform, dan biasanya disebut ulasan aplikasi. Ulasan yang diberikan berfungsi sebagai *feedback* bagi pengembang aplikasi dan membantu calon pengguna baru dalam mengambil keputusan apakah akan menggunakan aplikasi tersebut atau tidak.

2.2.2 Analisis Sentimen

Analisis sentimen atau *sentiment analysis*, adalah sebuah teknik dalam pemrosesan bahasa alami (*natural language processing* atau NLP) yang digunakan untuk mengidentifikasi dan mengukur emosi atau sentimen yang terkandung dalam teks. Proses ini bertujuan untuk menentukan polaritass dari sebuah teks yakni positif, negatif, atau netral. Analisa sentimen sering diimplementasikan ke berbagai jenis data, seperti ulasan produk, ulasan layanan, komentar di media sosial, dan media lain untuk memahami pandangan seseorang terhadap suatu brand, produk atau topik [38] [39].

2.2.3 Text Mining

Text mining adalah sebuah proses yang bertujuan untuk mengekstrak informasi berharga dari data teks yang tidak terstruktur. Proses ini menggabungkan teknik pemrosesan bahasa alami dan algoritma analisis untuk menemukan pola, tren dan wawasan yang tidak terlihat dalam data [40]. Text mining kerap digunakan ke berbagai bidang, termasuk analisa sentimen yang tujuan utamanya untuk memahami opini atau emosi yang terkandung dalam teks.

Preprocessing dalam teks mining adalah proses yang penting karena proses ini secara umum membersihkan teks yang akan diteliti. Adapun beberapa langkah dalam teks mining sebagai berikut [41]:

1. Pengumpulan data: mengumpulkan teks dari berbagai sumber, seperti ulasan produk, ulasan di *social media*, dan lain lain.
2. Pembersihan teks: menghapus karakter khusus, angka, emoji, hashtag, dan symbol.
3. *Tokenization*: memecah teks menjadi unit kecil(token)

4. *Remove Stopwords*: menghilangkan kata-kata umum yang tidak memberikan pengaruh signifikan (contoh: kata “dan”, atau”).
5. *Stemming*: Mengubah kata kata ke bentuk dasar (contoh: memasak menjadi ‘masak’)

2.2.4 Heuristic-based labeling

Heuristic based labeling atau biasanya dikenal sebagai *rule-based sentiment* adalah metode yang digunakan pada *natural language preprocessing* untuk memberikan label terhadap text berdasarkan aturan dan pola. Pola pada peraturan ini merujuk kepada struktur kata, frasa, dan penulisan yang menggambarkan indikasi positif, negatif atau netral[42]. Kelebihan menggunakan metode ini adalah transparansi karena aturan atau pola dapat diimplementasikan secara mudah sesuai dengan konteks. Heuristic based labeling juga dapat dikombinasikan dengan teknik *machine learning* ataupun *deep learning* dimana dapat menghasilkan hasil akurasi dalam menangani sentimen teks lebih detil [43].

2.2.5 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) adalah model *probabilistic generative* yang digunakan untuk menemukan struktur topik *latent* dalam kumpulan data, seperti corpus teks. Tujuan utamanya adalah untuk mengidentifikasi pola tersembunyi dalam data teks dengan mengasumsikan bahwa setiap dokumen merupakan campuran dari beberapa topik, dan setiap topik yang ada merupakan distribusi probabilistik atas kata kata dalam *corpus* [44]. Dikembangkan oleh David Blei, Andrew Ng, dan Michael Jordan pada tahun 2003, LDA menjadi salah satu teknik utama dalam pemodelan topik. Proses generatif LDA mencakup beberapa tahapan utama [45]:

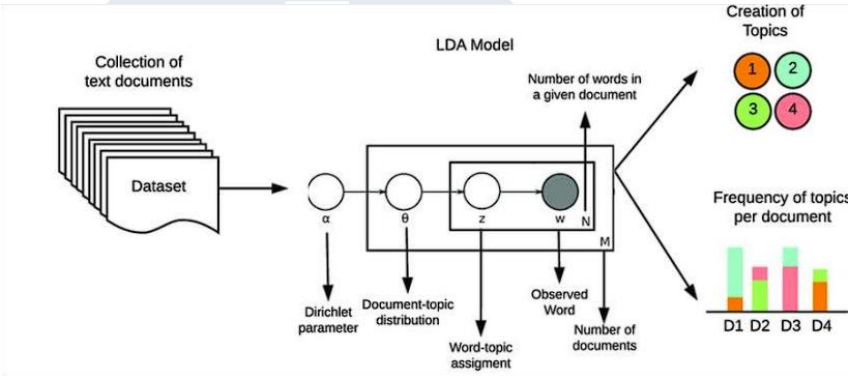
1. Pemilihan distribusi topik
   setiap dokumen memiliki distribusi topik tertentu yang diperoleh dari distribusi Dirichlet dengan parameter α. Distribusi ini menentukan proporsi berbagai topik dalam dokumen tersebut.

2. Pemilihan topik untuk setiap kata

   Untuk setiap kata dalam dokumen, topik dipilih berdasarkan distribusi topik yang telah ditentukan sebelumnya.

3. Pemilihan kata dari topik

   Setelah topik suatu kata ditentukan, kata spesifik dipilih berdasarkan distribusi kata dalam topik tersebut, yang dimodelkan menggunakan distribusi Dirichlet dengan parameter β.



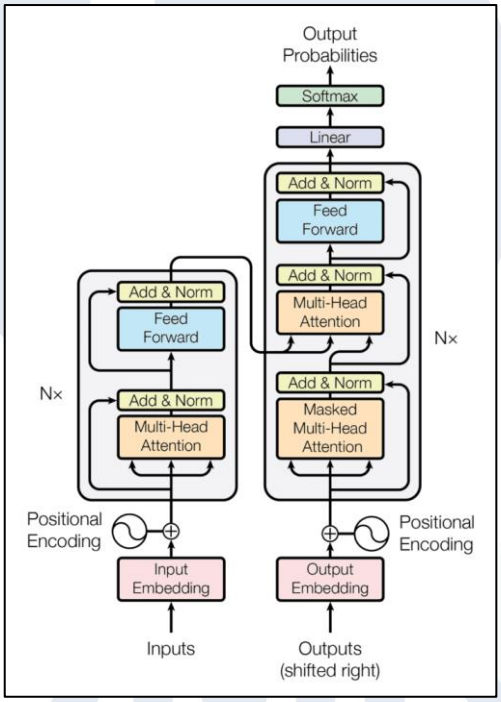
Gambar 2. 1 Komponen dari LDA [44]

Gambar 2. 1 merupakan komponen dari LDA dan secara matematis, adapun proses LDA dapat dirumuskan secara berikut [44]:

1. Untuk setiap topik K, dalam jumlah total K, distribusi kata ϕk ditentukan berdasarkan distribusi Dirichlet ϕk ~ Dirichlet(β)
2. Untuk setiap dokumen d dalam jumlah total, dilakukan langkah-langkah berikut:
   a. Distribusi topik θ_d diambil dari distribusi Dirichlet: θd ~ Dirichlet(α)
   b. Untuk setiap kata w_{dn} dalam dokumen d, dilakukan langkah-langkah berikut:
      1. Pemilihan topik Z_{dn} berdasarkan distribusi kategori yang ditentukan oleh θ_d: Z_{dn}~ Multinomial (θ_d)
      2. Pemilihan kata W_{dn}berdasarkan distribusi kata dalam topik Z_{dn}: W_{dn}~ $Multinomial$ (ϕZ_{dn}

2.2.6 Transformer Model

Transformer model adalah arsitekur *neural network* yang pertama kali diperkenalkan pada tahun 2017 dalam makalah berjudul "*Attention is All You Need*" [46]. Model ini dibuat untuk menangani data urutan seperti teks, suara, gambar, bahkan video dengan fokus utama pada efisiensi dan efektivitias hubungan antar elemen dalam sebuah *sequence*. Komponen utama Transformer terdiri dari *self-attention mechanism, encoder-decoder structure, positional encoding, feed forward neural network,* dan *multi-head attention*.



Gambar 2. 2 Transformer model Architecture [47]

Gambar menjelaskan bahwa *Input Embedding* (Encoder) intinya adalah kata kata yang diinput diubah menjadi *vector numeric*, sedangkan *positional encoding* menambahkan informasi posisi urutan kata, karena transformer tidak secara inheren memahami urutan seperti RNN. Encoder (kiri) bertugas untuk merepresentasikan konteks dari input dimana masing masing mempunyai tugas berikut [48].

1. *Multi head attention*
    a. Menganalisa hubungan antara kalimat dalam input

b. Menggunakan beberapa kepala(head) untuk menangkap hubungan dari berbagai aspek secara parallel.

2. *Add & Norm*
    a. Koneki residual (add) untuk menjaga informasi awal yang sudah diinput dari awal, dimana mencegah informasi yang hilang
    b. *Norm Layer* merupakan *normalization* menstabilkan proses pelatihan
3. *Feed forward neural network*
    a. Neural network untuk mengolah informasi lebih lanjut, dan komponen ini diulang sebanyak N=6 untuk menghasilkan representasi input dari layer, dan menambahkan *feature extractions.*

Pindah ke Gambar 2. 2 Decoder(kanan) bertugas untuk mengambil representasi dari encoder dan menghasilkan output dimana masing masing mempunyai tugas berikut [48]:

1. *Masked multi-head attention*

    Menganalisa kalimat seperti pada tahap encoder namun di tahap ini memblokir informasi dari kata-kata setelah posisi (masking), agar prediksi dapat dilakukan.

2. *Multi-head attention (Encoder-Decoder)*

    Menganalisa hubungan antara input (encoder) dan output kata yang sedang dibuat.

3. *Feed-forward Neural Network*

    Memiliki tugas sama seperti pada tahap encoder sebelumnya, yakni mengolah informasi lebih kompleks diulang sebanyak N = 6.

4. *Add dan Norm*

Memiliki tugas sama seperti encoder, menjaga stabilitas dan informasi awal pada saat pelatihan

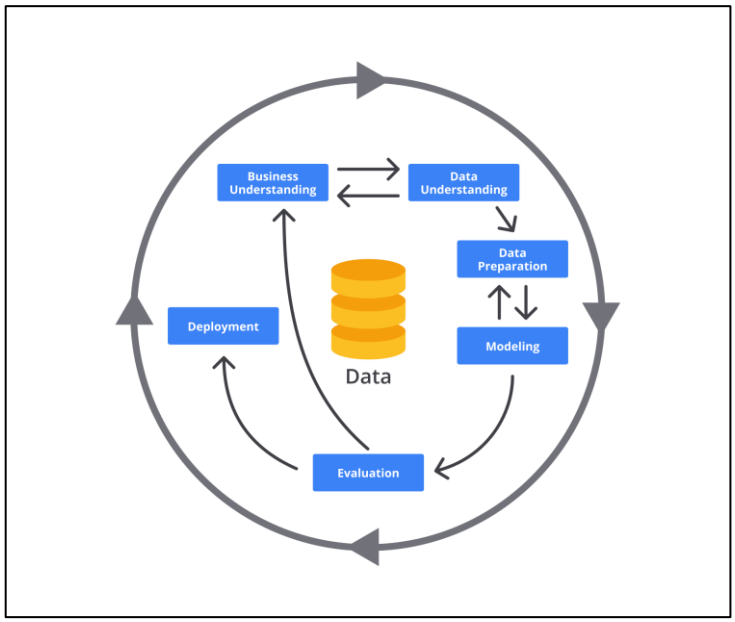
5. *Output embedding dan Softmax*
   a. Output dari seluruh rangkaian akan diubah menjadi representasi numerik, mengkonversi *hidden state* menjadi output yang akan diinginkan seperti "kata" yang akan prediksi.
   b. Softmax adalah fungsi dimana menghasilkan probabilitas, mengkonversi nilai output(logits) dan digunakan untuk membuat prediksi dari kata yang sudah dihasilkan berdasarkan labels.

2.3 Teori tentang *Framework*, Algoritma, dan Metode Evaluasi

2.3.1 Framework

2.3.1.1 CRISP-DM

Cross Industry Standard Process for Data Mining atau CRISP-DM adalah kerangka kerja standar yang digunakan untuk proses data mining. CRISP-DM juga merupakan salah satu metode yang banyak digunakan dalam penelitian *data science* [49].



Gambar 2. 3 Alur kerangka *CRISP-DM [50]*

Gambar 2.1 menggambarkan 6 tahapan tentang CRISP-DM yakni *business understanding, data understanding, data preparation, modeling, evaluation dan deployment* [51] . Berikut merupakan penjelasan dari masing-masing langkah:

1. *Business Understanding*

   Business understanding merupakan tahapan pertama dimana sebuah permasalahan bisnis dicari, tujuan bisnis yang diharapkan, beserta mencari solusi yang ada dalam sebuah permasalahan bisnis dengan penerapan *data science* ataupun *data mining.*

2. *Data Understanding*

   Data understanding merupakan tahap kedua setelah business understanding yang bertujuan untuk mencari dan mengumpulkan data penelitian yang sesuai baik dari internet, data buatan sendiri, maupun data internal dari sebuah perusahaan. Tidak hanya itu pada tahap ini peneliti juga melakukan eksplorasi data bisa berwujud statistika data maupun visualisasi data, dan melihat kualitas data yang akan digunakan.

3. *Data Preparation*

   Data preparation merupakan tahap ketiga setelah melewati tahap data understanding. Tahap ini tidak kalah penting karena didalamnya meliputi perbaikan data, pengecekan *missing data* (NaN), normalisasi data, *balancing data*, *split data*, dan lain lain. Tahap ini harus dilakukan dengan baik supaya pada saat data tersebut dijadikan model penelitian hasil performanya sesuai.

4. *Modeling*

   Modeling merupakan tahap ke empat setelah *data preparation*. Tahap ini melakukan pemodelan data, melibatkan pemilihan teknik pemodelan, pembangunan kasus uji, dan pembuatan model. Pada dasarnya, semua teknik data mining dapat digunakan dalam tahap ini. Keputusan tentang teknik yang dipilih cenderung bergantung pada masalah bisnis yang dihadapi dan karakteristik data yang tersedia.

Terutama bagaimana peneliti menjelaskan alasan di balik pemilihan model tersebut. Ketika membangun model, perlu menyesuaikan parameter-parameter tertentu sehingga model yang dibuat berjalan dengan lancar sesuai rencana.

5. *Evaluation*

   Evaluation merupakan tahap setelah *modeling* dimana pada tahap ini melakukan evaluasi terhadap hasil model yang sudah dibentuk. Apabila ada tidak kesesuaian dengan apa yang diharapkan pada tahapan *business understanding* maka perlu dilakukan uji ulang model dengan pendekatan yang berbeda, contohnya dalam hal ini mengatur parameter lain, jumlah pengulangan *training*, regularisasi dari setiap model yang ada.

6. *Deployment*

   Deployment merupakan tahap terakhir dimana pada tahap ini dilakukan implementasi model yang paling baik dan sudah dibuat pada tahap sebelumnya untuk dilakukan uji. *deployment* bisa berbentuk laporan akhir atau komponen perangkat lunak.

2.3.1.2 Optuna

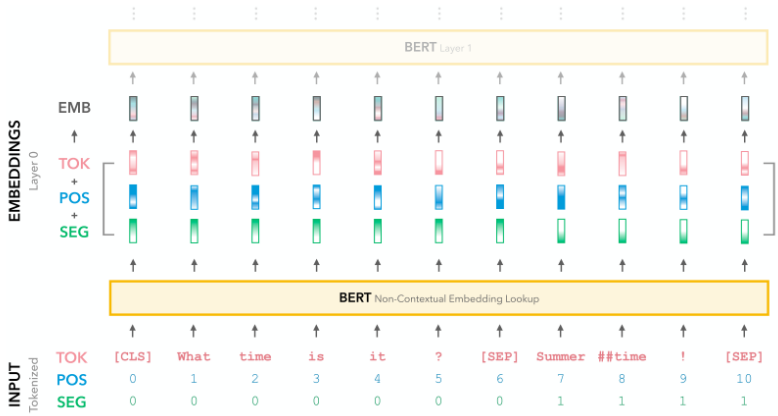
Optuna adalah sebuah *framework opensource* yang dikembangkan secara otomasi dengan tujuan mencari *hyperparameter* terbaik didesain terhadap proses *tuning parameter* untuk *model machine learning*. *Framework* ini memungkinkan pengembang menemukan hyperparameter yang optimal secara efisien dan efektif menggunakan metode optimasi berbasis Bayesian dengan teknik *Tree Parzen Estimator* (TPE) dan bisa diintegrasikan secara mudah menggunakan library machine learning seperti Pytorch, Tensorflow, XGBoost, LightBGM, dan scikit-learn [52].

2.3.2 Algoritma

2.3.2.1 BERT

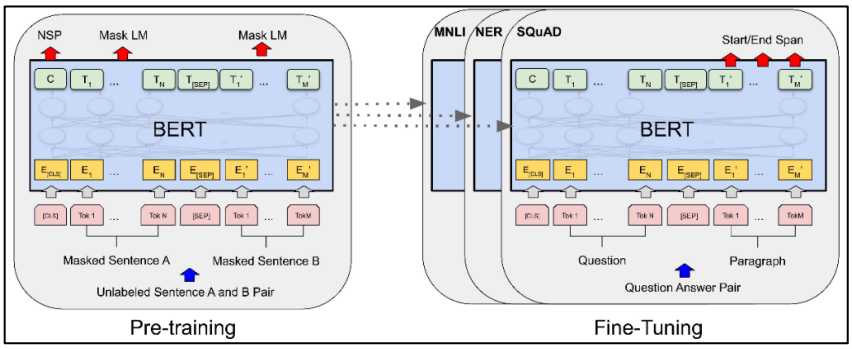
BERT (*Bidirectional Encoder Representations from Transformers*) adalah model bahasa pemrosesan alami (*Natural Language Processing*) yang dikembangkan oleh Google pada tahun 2018 untuk melakukan tugas pemrosesan bahasa alami [53]. BERT memiliki banyak model yang biasa

yaitu BERT base dan BERT large. BERT base mempunyai 12 *layers transformers block* dengan *hidden size* 768, dan *self-attention* heads 16. Cara kerja BERT secara keseluruhan adalah input yang dilakukan *sequence*, dimana BERT menggunakan *classification token* [CLS] dan *separate token* [SEP] untuk memahami apa yang diinput. Token [SEP] juga membantu memahami model di akhir input dan memulai *sequence* selanjutnya. Kemudian *token embedding* BERT menggunakan *positional encoding* [POS] dan *segment embedding* untuk setiap token [54].



Gambar 2. 4 Gambar representasi input cara kerja tokenisasi BERT [54]

Position embeddings menyimpan informasi mengenai posisi *token sequence*. *Segment embeddings* membantu model pada saat sedang input berpasangan. Kalimat pada token pertama akan didefinisikan embedding 0 sedangkan token kalimat kedua akan didefinisikan 1 sebagai *segment embeddings*.



*Gambar 2. 5 Bagan BE*RT yang terpecah ke tahap *pre training* dan *fine tuning [55]*

Selanjutnya, BERT mempunyai 2 tahap perancangan dengan tugas yang berbeda, yakni tahap *pre-training* dan tahap *fine-tuning*. Pada gambar 2. 5 *tahap pre-training*, data yang tanpa label akan dilatih dengan metode *unsupervised pre-training* yakni *masked language model* dan *next sentence prediction*. Kemudian, pada tahap fine tuning model yang sudah di latih sebelumnya akan disesuaikan supaya kedepannya dapat melakukan klasifikasi.

a. *Pre-training*

   Tahap ini representasi BERT akan dibentuk dari teks. Representasi teks kemudian akan diproses dari dua arah yang berbeda, dengan metode *masked language model* (MLM) dari arah kiri ke kanan dan kanan ke kiri. Pada *Masked Language Model*, sebanyak 15% token dalam teks masukan akan diganti dengan token khusus [MASK]. Model kemudian ditugaskan untuk memprediksi token asli yang disembunyikan tersebut berdasarkan konteks dari kedua arah. Tahap selanjutnya adalah, *Next sentence prediction* yang dirancang untuk melatih model mengenali hubungan antar kalimat [56]. Pasangan kalimat dimasukkan, dan model memprediksi apakah kalimat kedua merupakan kelanjutan logis dari kalimat pertama atau tidak. Kombinasi dari kedua tugas ini membantu BERT membangun representasi bahasa yang kaya dan fleksibel untuk berbagai aplikasi NLP.

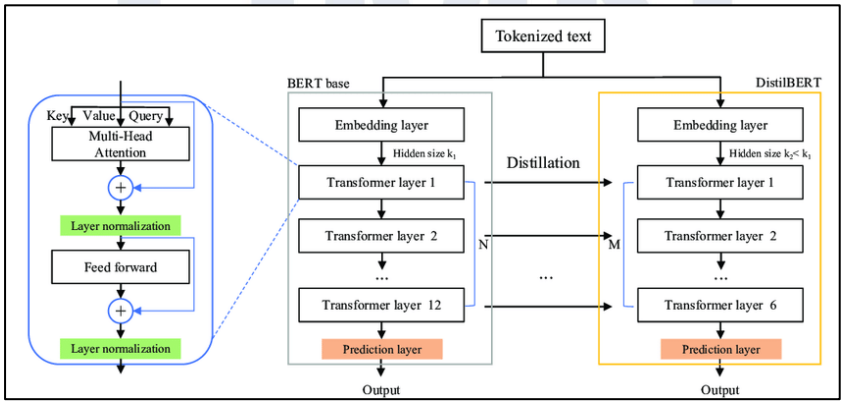
b. *Fine-tuning*

   Tahap fine tuning, adalah tahap dimana BERT melakukan penyesuaian model yang sudah di *pre-train* untuk ugas-tugas spesifik menggunakan dataset yang sudah diberi label. Pada tahap ini, parameter yang telah dilatih selama pretraining akan diperbarui agar model dapat menangani tugas tertentu, contohnya klasifikasi teks. MNLI (*Multi Genre Natural Language*) bertugas menginferensi hubungan logis antara dua kalimat dimana prosesnya adalah model menerima sepasang kalimat dan memutus

jenis hubungan di keduanya. NER (*Named Entity Recognition*) bertugas untuk mengidentifikasi entity, dan model diadaptasikan untuk melakukan prediksi label pada setiap representasi token yang tersedia dilabel dengan [CLS].

2.3.2.2 DISTILBERT

DistilBERT merupakan model pra-terlatih yang dikembangkan melalui penerapan teknik distilasi pengetahuan pada model BERT yang lebih besar. Secara arsitektur, DistilBERT memiliki kemiripan dengan BERT, di mana keduanya menggunakan transformator, khususnya di bagian blok encoder. *Transformator* sendiri terdiri atas *encoder* dan *decoder*. Salah satu perbedaan utama antara keduanya terletak pada penghapusan *pooling* dan penyematan berbagai jenis token pada DistilBERT [57]. Selain itu, model ini memanfaatkan dataset berukuran besar dengan dukungan akumulasi gradien selama proses pelatihan. DistilBERT juga menggantikan masking statis yang digunakan dalam BERT asli dengan masking dinamis dan menghilangkan tujuan pelatihan untuk prediksi kalimat selanjutnya (*Next Sentence Prediction* atau *NSP*). Jumlah lapisan encoder dalam BERT dasar, yang semula terdiri atas 12 lapisan, telah dikurangi menjadi 6 lapisan pada DistilBERT. Akibatnya, model ini memiliki parameter yang lebih sedikit, yaitu sekitar 66 juta parameter dibandingkan BERT biasa.



Gambar 2. 6 Arsitektur *model DistilBERT* dan komponennya [58]

Pada gambar diatas proses ini bertujuan untuk menghasilkan model yang lebih ringan dan efisien tanpa mengorbankan performa secara signifikan. Pada

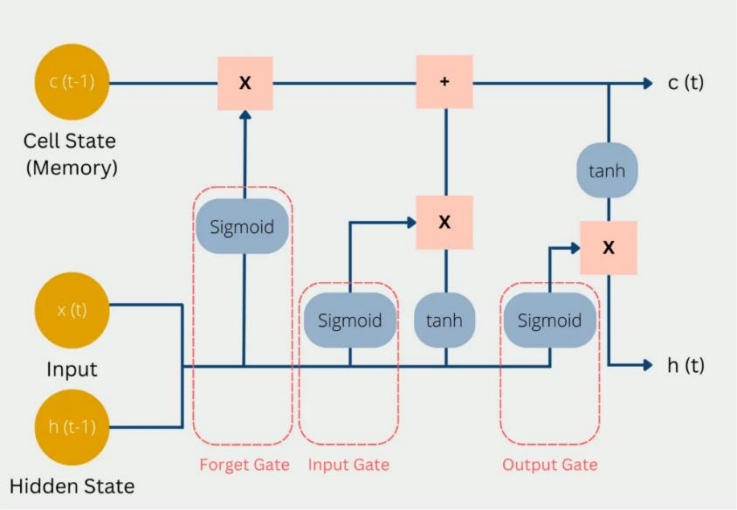
sisi kiri, terdapat model BERT Base yang terdiri dari beberapa komponen, yakni *Embedding Layer* yang berfungsi untuk mengonversi teks yang telah ditokenisasi menjadi representasi vektor, 12 lapisan Transformer dengan ukuran hidden sebesar k1, serta Prediction Layer yang menghasilkan prediksi berdasarkan representasi akhir dari lapisan transformer. Di sisi kanan, terdapat model DistilBERT, yang merupakan versi lebih ringan dari BERT. DistilBERT memiliki struktur yang serupa dengan BERT, namun menggunakan ukuran hidden k2 yang lebih kecil dibandingkan k1, hanya memiliki 6 lapisan Transformer, dan dilengkapi dengan *Prediction Layer* yang menghasilkan output prediksi yang setara dengan model BERT Base. Proses distilasi merupakan inti dari diagram ini, di mana DistilBERT dilatih dengan memanfaatkan pengetahuan dari BERT Base melalui pendekatan guru-murid. Dalam proses ini, model yang lebih kecil (DistilBERT) diarahkan untuk meniru hasil yang diperoleh dari model yang lebih besar (BERT Base) dengan mentransfer informasi penting dari lapisan-lapisan transformator BERT Base ke lapisan-lapisan DistilBERT. Untuk meningkatkan kualitas pembelajaran model DistilBERT, proses ini sering melibatkan penggunaan fungsi *loss* tambahan, seperti *soft loss*. Tujuan utama distilasi adalah untuk meningkatkan efisiensi model dengan mempercepat proses inferensi, mengingat DistilBERT hanya menggunakan enam lapisan dibandingkan dengan 12 lapisan pada BERT Base, serta mengurangi ukuran model dan kebutuhan komputasi, sambil tetap mempertahankan sebagian besar kemampuan pemahaman bahasa yang dimiliki oleh BERT. DistilBERT sering diterapkan dalam aplikasi yang memerlukan kecepatan tinggi dan penggunaan memori yang rendah, seperti inferensi waktu nyata pada perangkat dengan keterbatasan.

a. *DistilBERT-multilingual-cased by huggingface*

   DistilBERT *multilingual* merupakan model DistilBERT yang dikembangkan oleh komunitas pengembang distilbert pada platform *huggingface*. Model ini telah melakukan training dari Wikipedia dengan 104 bahasa yang berbeda. Model ini mempunyai 6 layers, 768 dimensions dan 12 *heads*.

2.3.2.3 LSTM

LSTM atau yang biasa disebut (*Long-Short Term Memory*) merupakan model algoritma *neural network* yang dibuat sebagai pengembangan dari *Recurrent Neural Network* untuk mengatasi keterbatasan RNN dalam menangani data yang sifatnya berjangka panjang pada *data time series*. LSTM dikembangkan untuk mempertahankan informasi krusial dalam periode waktu yang panjang serta secara efektif mengatasi tantangan *vanishing gradient* yang kerap dihadapi oleh RNN ketika memproses pelatihan pada data sekuensial berdurasi panjang [59]. Melalui kemampuannya ini, LSTM tidak hanya berhasil merepresentasikan keterkaitan temporal yang kompleks, tetapi juga memastikan bahwa informasi penting tetap tersedia untuk menunjang akurasi prediksi di masa mendatang [60][61]. Adapun struktur atau komponen utama dari LSTM adalah sebagai berikut[62]:



Gambar 2. 7 Arsitektur *model LSTM dan komponennya[62]*

a. Memory Cell

Komponen ini bertugas untuk menyimpan informasi dalam waktu jangka panjang. Memory cell ini memungkinkan LSTM untuk menyimpan data penting yang diperlukan untuk jangka panjang.

b. Forget Gate

Komponen ini bertugas untuk mengontrol informasi mana yang perlu dibuang dari *memory cell*. Gate ini menggunakan aktivasi sigmoid untuk menghasilkan nilai antara 0 dan 1, dimana 0 berarti

"forget" dan 1 "remember", dengan demikian LSTM dapat menghapus informasi yang tidak relevan secara efektif. Adapun rumus dari *Forget Gate* sebagai berikut [63]:

$$f_t = \sigma(W_f \cdot \left[h_{\{t-1\},} x_t\right] + b_f)$$

Rumus 2. 1 Rumus dari *Forget Gate* LSTM

c. Input Gate

Komponen ini bertugas untuk menentukan informasi baru apa saja yang akan ditambahkan ke dalam sel memory. Gate ini juga menggunakan fungsi sigmoid untuk mengatur seberapa besar informasi yang masuk dan yang akan disimpan, dikombinasikan dengan nilai yang dihasilkan oleh fungsi tanh untuk menyesuaikan skala data baru tersebut. Adapun rumus dari *Input Gate* sebagai berikut:

$$i_t = \sigma(W_i \cdot \left[h_{\{t-1\},} x_t\right] + b_i$$

Rumus 2. 2 Rumus dari *Input Gate* LSTM

d. Output Gate

Komponen ini bertugas untuk mengatur informasi yang akan dikeluarkan dari sel memori sebagai output pada setiap step. Output ini juga digunakan sebagai hidden state yang diteruskan ke langkah berikutnya dalam jaringan. Adapun rumus dari *Output Gate* sebagai berikut:

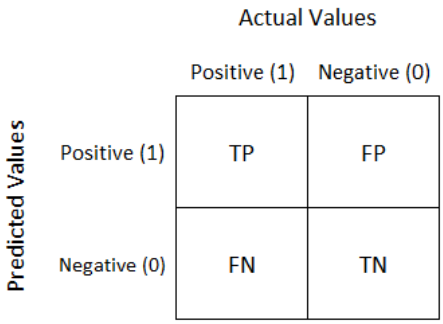
$$o_t = \sigma(W_o \cdot \left[h_{\{t-1\},} x_t\right] + b_o$$

Rumus 2. 3 Rumus dari *Output Gate* LSTM

2.3.3 Metrik Evaluasi

Metrik evaluasi adalah metrik yang digunakan sebagai indikator performa dalam hal ini seperti klasifikasi, regresi dan lain lain. Metrik evaluasi ini berdasar pada penggambaran visual *confusion matrix*. *Confusion matrix* adalah matriks yang menunjukkan perbandingan nilai-nilai actual dan nilai- nilai hasil prediksi dari hasil training yang dibuat [64].

Adapun *Confusion matrix* digambarkan sebagai berikut:



Gambar 2. 8 Confusion Matrix [64]

Berdasarkan confusion matrix, dapat diperoleh informasi mengenai nilai *True Positive (TP), True Negative (TN), False Positive (FP), dan False Negative (FN)* sebagai kategori utama dalam evaluasi model klasifikasi. *True Positive (TP)* merujuk pada jumlah sampel yang diprediksi sebagai kategori positif dan memang benar termasuk dalam kategori tersebut, selanjutnya *false positive* (FP) menunjukkan jumlah data yang diprediksi sebagai positif tetapi sebenarnya negatif. Sementara itu, true negative (TN) mencerminkan jumlah data yang diprediksi sebagai negatif dan memang benar negatif, dan false negative (FN) menggambarkan jumlah data yang diprediksi sebagai negatif tetapi sebenarnya positif.

2.3.3.1 Accuracy

Accuracy merupakan metrik yang menunjukkan tingkat keakuratan model dalam memprediksi label secara keseluruhan. *Nilai accuracy* dihitung dengan membandingkan jumlah elemen yang diprediksi dengan benar terhadap total elemen yang diprediksi. Berdasarkan *confusion matrix*, accuracy dapat diperoleh dengan menjumlahkan seluruh elemen pada diagonal utama, kemudian membaginya dengan jumlah keseluruhan elemen pada confusion matrix [65]. Adapun rumus dari accuracy sebagai berikut:

$$Accuracy = \frac{TN + TP}{TP + FP + TN + FN}$$

Rumus 2. 4 Rumus *Accuracy*

2.3.3.2 Precision

Precision merupakan metrik yang mengukur tingkat keberhasilan model dalam memprediksi nilai positif, serta sejauh mana model dapat dipercaya dalam menghasilkan prediksi positif. *Precision* dihitung dengan membandingkan jumlah elemen yang benar-benar positif terhadap total elemen yang diprediksi sebagai positif [65]. Adapun rumus dari *precision* sebagai berikut:

$$Precision = \frac{TP}{TP + FP}$$

Rumus 2. 5 Rumus *Precision*

2.3.3.3 F1-Score

F1-score merupakan metrik yang digunakan untuk mengevaluasi performa model dengan menghitung rata-rata harmonik antara precision dan recall. Metrik ini memberikan keseimbangan antara kedua indikator tersebut, sehingga dapat menggambarkan kinerja model secara lebih komprehensif, terutama ketika terdapat ketidakseimbangan antara jumlah data positif dan negative [65]. Adapun rumus dari *f1-score* sebagai Berikut:

$$F1\ Score = \frac{2TP}{2TP + FP + FN}$$

Rumus 2. 6 Rumus *f1-score*

2.3.3.4 Recall

Recall merupakan metrik yang mengukur tingkat keakuratan model dalam memprediksi elemen yang bernilai positif. Recall dihitung dengan membandingkan jumlah elemen yang diprediksi sebagai positif terhadap total elemen yang sebenarnya bernilai positif [65]. Adapun rumus dari *recall* sebagai berikut:

$$Recall = \frac{TP}{TP + FN}$$

Rumus 2. 7 Rumus *Recall*

2.4 *Tools* Penelitian

2.4.1 Python

Python adalah sebuah bahasa pemrograman tingkat tinggi yang sering digunakan dalam berbagai konteks seperti pengembangan perangkat lunak, analisis data, kecerdasan buatan (*artificial intelligence*), dan berbagai jenis komputasi lainnya. Bahasa pemrograman ini terkenal dengan sintaksis yang sederhana sehingga lebih mudah dibaca dan dipahami, sehingga banyak dipilih oleh pemula yang ingin memasuki dunia pemrograman [66].

Gambar 2. 9 Logo bahasa pemrograman *Python [67]*

Tidak hanya itu, Python juga mempunyai banyak library terutama dalam pengembangan *machine learning, deep learning, data analyst*, maupun *data science* contohnya Numpy, Tensorflow, Keras, PyTorch, Pandas, Seaborn, Sk-learn, dan lain lain [68]. Python juga memiliki beberapa keunggulan dibandingkan dengan Bahasa pemrograman lainnya yaitu [69]:

- Python adalah bahasa pemrograman yang efisien dalam artian penulisan kode tidak perlu berbelit belit seperti bahasa pemrograman lain contohnya Java dan C++. Tidak hanya itu Python juga bisa digunakan hampir ke semua *environment* sehingga user tidak perlu pusing ketika ingin membuat sebuah projek kodingan.
- Python banyak digunakan di bidang *machine learning* dan *big data* karena saat ini banyak penelitian menggunakan bahasa python.
- Python merupakan bahasa pemrograman yang *open source* dalam artian tidak ada limitasi ketika digunakan, dan orang- orang dapat melakukan kontribusi besar terhadap komunitas python baik dengan cara distribusi koding *open source*. Kemudian dukungan terhadap operating system selain Windows seperti LINUX, MAC OS juga bisa berjalan dengan baik ketika menjalankan bahasa Python [70].

2.4.2 Jupyter Notebook

Jupyter Notebook merupakan lingkungan komputasi interaktif yang dirancang untuk mendukung pembuatan serta kolaborasi dokumen yang mengintegrasikan kode pemrograman, teks deskriptif, visualisasi data, dan berbagai elemen interaktif lainnya [71]. Keunggulan dari Jupyter notebook adalah mampu menampilkan output hasil kode yang dijalankan dibawahnya, sehingga pengguna bisa melihat outputnya secara langsung tanpa perlu menjalankan seluruh codingan. Nama "Jupyter" merupakan singkatan yang berawal dari gabungan kata Julia, Python, dan R, yang menyediakan dukungan untuk berbagai platform bahasa pemrograman utama yang berperan penting dalam analisis data dan proses komputasi. Jupyter Notebook sangat populer di kalangan ilmuwan data, peneliti, dan developer karena kemampuannya untuk mengintegrasikan berbagai komponen dalam satu dokumen yang mudah dibaca dan dibagikan. Jupyter notebook juga tidak terbatas dengan kode dan teks tapi bisa menginput widget lainnya seperti gambar, video dan menggabungkan elemen dari HTML dan JavaScript [72].

Gambar 2. 10 Logo *Jupyter Notebook [73]*

2.4.3 Visual Studio Code

Visual Studio Code adalah sebuah perangkat lingkungan pengembangan terpadu (IDE) bersifat *open-source* yang telah dikembangkan oleh *Microsoft corporation*. Visual Studio Code diciptakan untuk menyediakan pengembang dengan perangkat yang kuat untuk membuat, mengubah, dan mengelola kode yang sudah ada di dalam berbagai bahasa pemrograman. Visual studio code juga merupakan software yang *open source* dan gratis sehingga bisa digunakan oleh siapa saja [74]. Tidak hanya itu Visual Studio code juga ada *tools* untuk melakukan integrasi dengan aplikasi ketiga guna untuk mendukung aktivitas coder pada saat sesi ngoding, kolaborasi dengan pihak lain dengan *tools* tambahan seperti Git, dan Github. Tidak hanya itu Visual studio code sekarang juga bisa melakukan fungsi copilot sehingga memudahkan coder agar mengetik

code dengan menggunakan beberapa prompt saja dengan integrasi lewat Github copilot [75]. Visual studio code juga menyediakan banyak extension tambahan sehingga *user* bisa melakukan kustomisasi tampilan bahkan menambah tools diluar VS code mereka. Visual studio code juga sudah mendukung penuh segala perubahan kualitas yang dilakukan seperti *log* melalui situs resmi Github mereka sehingga user bisa melakukan sharing code bahkan diskusi antar user [76].

Gambar 2. 11 Logo *Visual Studio Code [77]*

2.4.4 HuggingFace

Huggingface adalah sebuah platform dimana komunitas *machine learning* berkumpul bersama untuk membuat *models*, *datasets* dan application. Huggingface berdiri pada tahun 2016 yang dikembangkan oleh Clément Delangue, Julien Chaumond, dan Thomas Wolf. Huggingface memiliki banyak model yang sudah di publish, dan diantaranya adalah transformer library dimana mendukung beberapa framework *machine learning* antara lain Tensorflow, dan PyTorch [78]. Selain itu *huggingface* juga memiliki *hub* yang memungkinkan pengguna membagikan model yang dimodifikasi serta meluncurkan sebuah aplikasi website sederhana di web *huggingface* hasil dari *machine learning* agar menjadi pembelajaran bersama dengan pengguna lainnya.

Gambar 2. 12 Logo *HuggingFace [79]*

2.4.5 Streamlit

Streamlit merupakan *framework open-source* berdasarkan pada bahasa pemrograman Python yang digunakan untuk mengembangkan *website* interaktif sederhana, intuitif, dan cepat dengan tujuan utama membangun aplikasi web pada bidang *data science*, *data analyst*, dan *machine* [80]. *Framework* ini juga populer dikalangan *data scientist*, dan developer karena pengaplikasiannya tergolong mudah dalam membuat UI yang *responsive* dan tanpa perlu memahami dasar pembuatan website. Kunci Fitur streamlit antara lain [81]:

1. Mudah digunakan

   Streamlit hanya perlu menggunakan beberapa garis code terintegrasi dengan API
2. *Data visualization*

   Streamlit bisa mengintegrasikan library visualisasi seperti Seaborn, Matplotlib, dan plotly.
3. *Interactive widgets*

   Terdapat widgets seperti slider, button, text inputs supaya tampilan web lebih interaktif.
4. *Realtime update*

   Streamlit dapat secara otomatis mengupdate isi dari apa yang dijalankan, dan bila ada perubahan maka perubahan tersebut akan diubah.
5. *Deployment Ready*

   Streamlit dapat di *deploy* ke banyak platform besar seperti Heroku, AWS dan lain lain yang menjadi salah satu platform yang kuat untuk membangun aplikasi.

Gambar 2. 13 Logo dari *Streamlit* [82]

2.4.6 Windows

Windows adalah suatu jenis sistem operasi perangkat lunak yang dirancang oleh Microsoft corporation. Perangkat lunak ini berperan sebagai perantara atau penghubung antara perangkat keras komputer, seperti CPU (*Central processing unit*), memori, GUI, dan perangkat input/output, dengan aplikasi perangkat lunak yang dijalankan di dalam komputer [83]. Windows juga terkenal populer dikalangan pengguna baik orang awam dan developer karena UI nya yang friendly dan banyak aplikasi yang mendukung sehingga menjadi operasi sistem perangkat lunak yang paling sering digunakan oleh orang orang.

Gambar 2. 14 Logo *Windows* [84]